International Journal of Exploring Emerging Trends in Engineering (IJEETE)
Vol. 03, Issue 02, Mar-Apr, 2016 Pg. 102 - 106 WWW.IJEETE.COM

"SMART-SHARE"- FASTEST FILE SHARING APPLICATION

SWAPNIL SONAWANE

Assistant Professor, Department of Computer Engineering
Vidyalankar Institute of Technology, Mumbai, India

ABSTRACT - *In all modern smart phones there are various applications (apps) which can be used for transferring or sharing text files, images, audios, videos etc. But the main disadvantage of these applications is, they cannot be used to send and receive compressed files. Using "SMART-SHARE", it is possible to send and receive all kind of files from sender mobile to recipient mobile in compressed form for faster data transfer.*

Keywords— Huffman encoding, SHAREit, Lossless Compression, Base 64, AAC, HEVC, WiFi

I. INTRODUCTION

The Android is a growing technology which has started to fulfil various needs with lots of application (apps) to make things simple, easier and convenient to mobile end users. Many android devices can support different file transfer applications like SHAREit, Software data cable, Telegram etc. These applications are used to transfer all kind of files like text, PDF, audios, videos etc. Using these applications, we can transfer text, images, audios and videos.

In this android application, we can transfer text, images, audios and videos from one device to another in less amount of time, as we can transfer them in compressed form to reduce the data transfer time. Though wireless communication is more expensive in nature, we create an application with cost efficient communication over the wireless network. By using compression techniques, we reduce the data size of the file to transfer by using ad-hoc Wi-Fi connections.

II. PROBLEM DEFINITION

In different android applications like SHAREit, Software Data Cable, SuperBeam etc. the can transfer different file formats like documents, photos, music, videos, and even apps. Normally all these apps creates a virtual Wi-Fi hotspot from the sender phone, then it allows the receiver phone to connect, As it's a peer to peer connection, thus logical speed can get up to 10 MBPS. The Wi-Fi hotspot can transfer the data via Wi-Fi using the P2P connection and thus its very efficient to transfer data as the speed gets pity high. [1].

All these applications transfer files about 40 times faster than the traditional Bluetooth file transfer system, i.e. these apps transfer data very quickly, even the large files from one device to another, even in cross platform device also. Also many of these apps are available for different operating systems like android, windows, iOS etc. [3]

It is observed that we can transfer files using these apps in non-compressed form, we cannot transfer files such as text files, images, audios or videos in compressed form to save file transfer time.

The files that we transfer using any normal app, from one device to other, will transfer it by creating a Wi-Fi hotspot and then by establishing a P2P connection between two devices.

Using these we can transfer original file in uncompressed form, although it takes less time to transfer because of P2P connection, some files like high definition videos or high resolution images and long duration audios etc., it takes long time to transfer. Also it suffers some different problems like more processing power is required to send and receive file during the file transfer time, and it consumes more battery power during the file transformation etc. All these problems occur because of long file transfer time using different traditional apps. So the main demand is to have an app which can transfer different types of files in least possible time, which can be achieved using "SMART-SHARE" app, by transferring files in compressed form to reduce file transfer time.

III. COMPRESSION TECHNIQUES

Compression is mainly used because it helps to reduce the consumption of expensive resources, like bandwidth of transmission. On other side, compressed data should be decompressed and this extra processing may disturb other applications running in parallel. The design of data compression schemes hence involves trade-offs between different factors, including the compression degree, amount of distortion, and the computational resources which

are used to compress and decompress the data [2].

We studied various encoding techniques and select the Huffman technique for text file compression, Base64 encoding for image compression, AAC encoding for audio or mp3 compression and HEVC encoding for video compression.

A. *Huffman Encoding*

Huffman coding is an entropy encoding algorithm used for lossless data compression in information theory. The term refers to the use of a variable-length code table for encoding a source symbol, where the variable-length code table has been derived in a particular way based on the estimated probability of occurrence or frequency for each possible value of the source symbol [4].

In Huffman coding the characters in a data file are converted to a binary code, where the most common characters in the file have the shortest binary codes, and the least common characters have the longest binary codes.

Huffman encoding uses a strictly binary tree where each non leaf node has two children. The Huffman algorithm works as follows [4]

1) Creating the tree:

1. Start with as many leaves as there are symbols.
2. Enqueue all leaf nodes into the first queue (by probability in increasing order so that the least likely item is in the head of the queue).
3. While there is more than one node in the queues:
   3.1. Dequeue the two nodes with the lowest weight.
   3.2. Create a new internal node, with the two just removed nodes as children (either node can be either child) and the sum of their weights as the new weight.
   3.3. Enqueue the new node into the rear of the second queue.
4. The remaining node is the root node; the tree has now been generated.

2) Code generation of each symbol:

[1] Start from the root node. For each down left traversal, add a '0' to the code and a '1' for each down right, add a '1'.
[2] When you reach a leaf node, the current code is the code for that character.
[3] When travelling to the parent of a node, delete the last added bit from the code.[1]

B. *Base 64 Encoding*

Base64 is a group of similar binary-to-text encoding schemes that represent binary data in an ASCII string format by translating it into a radix-64 representation.

The goal here is to convert/transform Image into Base64 String and convert/transform Base64 String back to Image without compromising an image data.

The Bese64 encoder works as follows [6]:

1. Prepare a byte array from image to be encoded
2. Once we have byte array of Image file, apply below method to convert byte array into Base64 string using **Base64.encodeBase64URLSafeString()**
3. Compress that string using Huffman Encoding, and transfer that encoded text to recipient
4. Decompress the encoded text using Huffman Decoder to get the plain text i.e. Base64 string
5. Once we have Base64 string, apply below method to convert string into byte array using **Base64.decodeBase64()**
6. Convert byte array into an image to retrieve original image

C. *AAC Encoding*

Advanced Audio Coding (AAC) is a standardized, lossy compression and encoding scheme for digital audio. Designed to be the successor of the MP3 format, AAC generally achieves better sound quality than MP3 at similar bit rates [7].

The AAC encoder works as follows:

The AAC encoder performs coding of the low frequency components of the input sound with the conventional AAC encoder.

The AAC encoder transforms the input sound into a frequency spectrum by using a Modified Discrete Cosine Transform (MDCT) followed by quantization. Also, based on a psychoacoustic analysis, the perceptual importance of each frequency is determined so that a larger number of quantization bits can be allocated to more important frequencies and the number of quantization bits for less important frequencies can be reduced to

enhance the coding efficiency. Further, to improve the coding efficiency, optional coding tools are available such as a block switching tool, a Temporal Noise Shaping (TNS) tool, and a Mid/Side (MS) stereo tool[8][9].

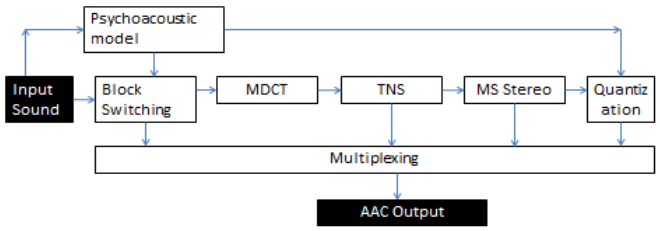

Figure 3-1: AAC Encoder

The decoders for both Layer-3 and AAC are fully specified in the relevant ISO standards, this makes listening tests of MP3 or AAC decoders a moot exercise:.

D. HEVC Encoding

HEVC is said to double the data compression ratio compared to H.264/MPEG-4 AVC at the same level of video quality. It can alternatively be used to provide substantially improved video quality at the same bit rate. It can support 8K UHD and resolutions up to 8192×4320[10].
The HEVC (H.265) encoder works as follows:

It looks at multiple frames to see what doesn't change. In most scenes in a TV show or movie, the vast majority of the frame doesn't change much. Think of a scene with someone talking. The shot is mostly their head. For that matter, most of the pixels representing their face probably won't change much (other than their lips, of course). So instead of encoding every pixel from every frame, an initial frame is encoded, and then after that only what changes is encoded. [11]

HEVC then expands the size of the area that's looked at for these changes. Larger and smaller "blocks" essentially, which offers additional efficiency. Those can be bigger, smaller, and differently shaped with HEVC, larger blocks, for example, were found to be more efficient. [12]

HEVC re-uses many of the concept defined in H.264. Both are block based video encoding techniques so have the same roots and the same approach to encoding:
1. subdivision of picture in macro blocks, eventually sub-divided in blocks
2. reduction of spatial redundancy using intra-frame compression techniques
3. reduction of temporal redundancy using inter-frame compression techniques (motion estimation and compensation)
4. residual data compression using transformation & quantization
5. reduction of final redundancy in residuals and motion vectors transmission and signaling using entropy coding [13]

So we selected these mentioned compression or encoding algorithms to compress text, images, audio and video as the aim of our system is to improve best case compressions without affecting the quality of the original file

In our proposed method we create an application with file compression over the WiFi network connection. We implement the Huffman, Base64, AAC and HEVC encoding for the file compression and send the compressed file to the recipient device. We will transfer it by creating a Wi-Fi hotspot and then by establishing a P2P connection between two devices.

Following are the various activities used in SMART-SHARE application:

1. *User select a file*
   In this, the user of the system browses and selects a file on his/her mobile phone, which he/she wants to compress and send it to the receiver.
2. *File passes through compression algorithm*
   In this, the selected file, depending upon the file type i.e. either text, image, audio or video will go under the compression or encoding algorithm and the encoded code will get generated by the algorithm
3. *Encoded code sent after compression to receiver*
   After compression of the file, the generated encoded code will get sent to the recipient of the file
4. *Encoded code passes through decompression*
   At recipient end, the encoded code will undergo decompression process to produce the original file
5. *File stored at receiver end*

After decompression the original plain file will get saved at the recipient device.

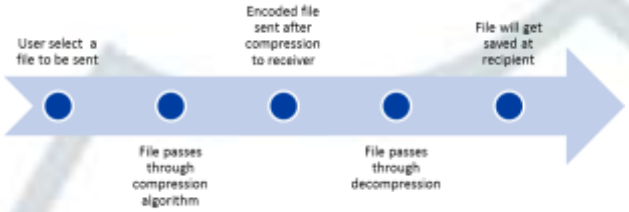

Figure 3-2: Process diagram of application

Following are the different modules which can be used while creating this application:

1. *Application Design*

This module is used to provide the design of the application, User interface is the major factor for an application to get an successful attention. It should be user friendly to cover the attention.

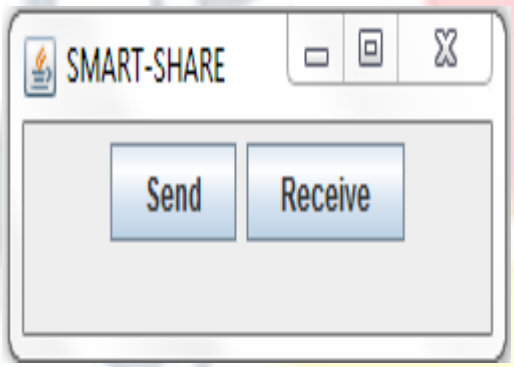

Figure 3-3: Sending or receiving file

2. *Pairing Device*

In this module we pair two devices so that we will transfer file by creating a Wi-Fi hotspot and then by establishing a P2P connection between two. Once the devices are connected then the file is transferred over the connection.

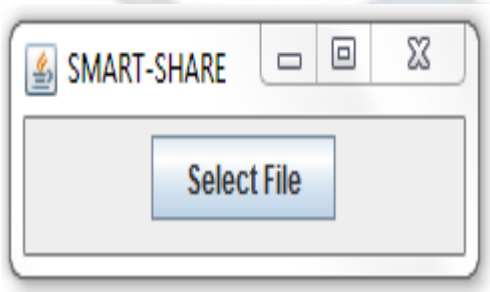

Figure 3-4: Selecting or browsing a file to send

3. *File Compression*

In this module we use different encoding techniques to compress the desired file.

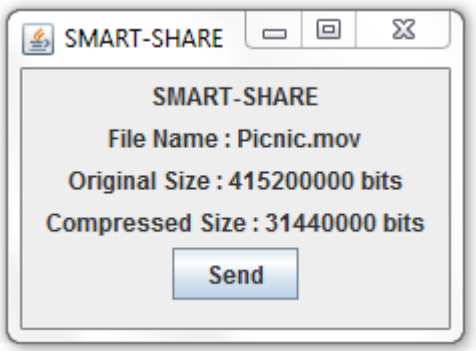

Figure 3-5: Sending file after compression

4. *Data Streaming*

In this module we stream the compressed data to the receiver which is already paired with sender. The receiver device, on receiving the compressed data, decompresses the data to its original file and save it to recipient device

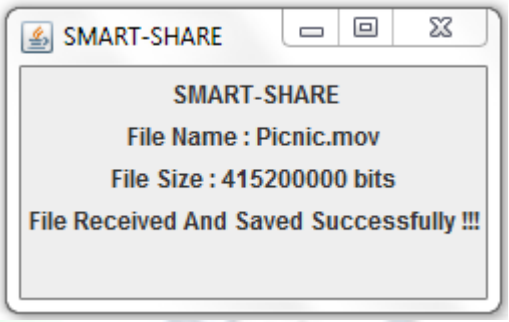

Figure 3-6: Receiving file after compression

IV. RESULTS ACHIEVED

To prepare the encoded file, we studied over different text messages, text files, images, videos, researches and technical papers. Using these statistics, we observed that Huffman, Base64, AAC and HEVC encoding are best encoding techniques to compress text, images, audio and video files and send compressed files from one mobile to another

It is observed that using Huffman, we will get around 50-60 per cent compression on text files using Base64, we can get around 80 per cent compression, using AAC encoding we can get around 60 per cent compression and using HEVC encoding we can get around 50 per cent compression

Table 4-1:

File Name	Input File Size (In bits)	Output File Size (In bits)
Example1.doc	68096	29433
Example2.doc	58880	23640
Example3.doc	83968	46876
Example4.doc	20480	4836
Example5.doc	27648	10921
Pict3.bmp	1440054	276506
Pict4.bmp	1440054	282824
Pict5.bmp	1440054	318178
Pict6.bmp	1365318	366830
Kalimba.mp3	64160000	432000000
Maid with the Flaxen Hair.mp3	31360000	20000000
Sleep away.mp3	36880000	24800000
Wildlife.wmv	200000000	55520000
Picnic.mov	415200000	31440000

V. APPLICATIONS

The most useful and common application of our research extends to the field of mobile computing. An android application can easily be developed which will prove immensely popular to the end user since it saves file transfer time, because compressed text, images, audios and videos can get transferred faster than their corresponding original files.

VI. CONCLUSIONS

In today's world, many projects and applications are being developed to overcome boundaries different types of file sharing. While incredible progress has been made in the field of mobile communication, support for increasing various end user demands remains a concern. Various advanced techniques are necessary in order to meet the challenges of business. "Speedy-Share" comes up with a new model which can revolutionize the way file transfer can be done between individuals. It will help file transferring more faster and efficient and can bring a good boost in mobile industry.

VII. ACKNOWLEDGEMENT

We are indebted to all teaching and non-teaching faculties of Department of Computer Engineering, Vidyalankar Institute of Technology for their guidance and encouragement to write this technical paper. Their invaluable guidance and unconditional support motivated us to work hard towards achieving our desired goals.

REFERENCES

[1] https://www.quora.com/Android-Applications/How-does-the-Share-It-Android-app-work
[2] Mohammed Al-laham & Ibrahiem M. M. El Emary "Comparative study between various algorithms of data compression techniques" in Proceedings of the World Congress on Engineering and Computer Science 2007, WCECS 2007, October 24-26, 2007, San Francisco, USA
[3] http://shareit.lenovo.com/faqs.html
[4] S.R. Kodituwakku and U. S. Amarasinghe "Comparison of lossless data compression algorithms for text data" in Indian Journal of Computer Science and Engineering, Vol 1 No 4 416-425 ISSN : 0976-5166
[5] Mamta Sharma "Compression Using Huffman Coding" in International Journal of Computer Science and Network Security, VOL.10 No.5, May 2010
[6] http://myjeeva.com/convert-image-to-string-and-string-to-image-in-java.html
[7] http://www.winxdvd.com/resource/aac.htm
[8] http://www.fujitsu.com/global/documents/about/resources/publications/fstj/archives/vol44-3/paper19.pdf
[9] https://graphics.ethz.ch/teaching/mmcom12/slides/mp3_and_aac_brandenburg.pdf
[10] https://en.wikipedia.org/w/index.php?title=High_Efficiency_Video_Coding&redirect=no
[11] http://www.cnet.com/news/what-is-hevc-high-efficiency-video-coding-h-265-and-4k-compression-explained/
[12] http://www.cnet.com/news/what-is-hevc-high-efficiency-video-coding-h-265-and-4k-compression-explained/
[13] https://sonnati.wordpress.com/2014/06/20/h265-part-i-technical-overview/

VIII. AUTHOR'S BIBLIOGRAPHY

Mr. Swapnil Sonawane received is M.E. degree in Information Technology from Vidyalankar Institute of Technology in year 2015. Presently he is working as Assistant Professor in Department of Computer Engineering in Vidyalankar Institute of Technology

